Amendments to the Drawings:

Provided herewith under separate cover are corrections to figures 1, 4, 7, 9, 18, and 19 to add reference labels.

REMARKS

This Amendment is filed in response to the Office Action dated January 25, 2006. Applicants first note with appreciation the Examiner's thorough examination of the application. In light of the Office Action, Applicants have amended the figures and specification. Applicants have also canceled claims 2, 25, 34, 37, 44, and 48 and amended claims 1, 3-5, 8-10, 12-18, 20-24, 26-33, 35, 36, 38-43, 45-47, 49-53, and 55. Applicants respectfully submit that the application as amended is patentable and respectfully requests reconsideration and allowance of the application in light of the following remarks.

I. Requested Copy of Document

On page 2, the Office Action requests a copy of document D2: XP10094549. Applicants have submitted herewith a copy of the reference.

II. The Drawings Are In Proper Form

On page 2, the Office Action objects to Figure 1, 4, 7, 9, 18, and 19 as lacking reference labels. Applicants submit herewith under separate cover amendments to these figures. Applicants have added reference numbers to the figures and amended the specification to include the added labels. Applicants respectfully submit that the drawings are now in proper form.

III. The Method Claims Are Proper and Supported

On page 2, the Office Action objects to method claims 1-23, 35-46, 48-49, and 51-55 because there is no flowchart accompanying these claims in the specification. Applicants respectfully submit that a flow chart is not needed for these claims. The content of the claims is recited in the summary of the invention and described in greater detail in the detailed specification. Applicants refer to MPEP § 601.01(f) which recites "It has been PTO practice to treat an application that contains at least one process or method claim as an application for which a drawing is not necessary for an understanding of the invention under 35 U.S.C. § 113 (first

sentence)." The present application clearly includes method claims, therefore, flow charts are not required.

IV. Claim 47 Recite Statutory Subject Matter

On page 3, the Office Action rejects claim 47 under 35 U.S.C. § 101 as lacking statutory subject matter. Applicants disagree with this rejection and believe that claim 47 as originally drafted was proper. However, Applicants submit that point is moot in light of the amendments made to claim 47.

V. The Claims Are in Proper Form

On page 3, paragraph 6, the Office Action rejects claims 1, 22-24 and 34 under 35 U.S.C. § 112, first paragraph as reciting only a single means. Applicant has amended these claims to include more than one means and respectfully submits that the claims are in proper form.

On pages 3-5, the Office Action rejects many of the claims under 35 U.S.C. § 112, second paragraph as indefinite. Applicants have addressed many of these rejections via claim amendments. Applicant, however, would like to comment on some of the rejections.

Claim 16, as amended, recites "wherein digitized ~~digital~~ outputs from the fault free and the faulty circuits are used to determine the optimum digitized ~~digital~~ input test signal." Applicants respectfully submit that the claims are clear that output signals from the two circuits are used to determine an optimum input signal.

Claim 17, as amended, recites "a fault detection ratio ~~is used~~ to determine the digitized input test sequence, ~~optimum circuit, this~~ the fault detection ratio being defined as the proportion of a set of predefined faults that can be detected according to a set of criteria for fault discrimination. The claim clearly recites what the fault detection is.

On page 5, the Office Action rejects Claims 42-46, 48-49, and 53-55 as hybrid claims reciting both system and method recitations. Applicants respectfully disagree. These claims are computer program claims. While at first blush the recitations to be steps, if read with the preamble, it is clear that these are computer code instructions on a medium. The claims are in proper computer program product form.

VI. The Claims Are Patentable

The Office Action rejects the claims as anticipated by the Balivada article. The Office Action alleges that all aspects of the claims are disclosed in the article. Applicant respectfully disagrees.

The present application is directed to two different aspects of analogue or mixed-signal circuit testing. These are (a) the determination of an optimised input test signal and (b) the use of that optimised signal in a real circuit test environment. Claims 1 to 33 and 47 to 52 are directed to (a) and claims 35 to 46 and 51 to 55 are directed to (b).

The Balivada article describes an approach to test generation for parametric faults in linear analogue circuits. As noted on page 30, column 2, lines 4 to 8 of Balivada, this approach is very specific, and is directed to a class of mixed-signal circuits that have a linear analog front end and a digital filter macro. The basic mixed signal circuit is shown in Figure 1. This has an analog block and a digital block, with an ADC connected between them. In use, analog signals are applied to the analog block, passed through an analogue to digital converter and into the digital block.

To detect faults for a circuit of the general form of Figure 1, the Balivada article discloses a simulation technique. As noted in section 4, in order to provide a common framework for the simulation of both the digital and analog circuits of Figure 1, the analog blocks are firstly transformed so that they can be represented in the sampled Z-domain. This means that for the purposes of the simulation the analog circuit is effectively transformed into an equivalent digital representation. Once this is done, a digital signal test simulation is run to identify specific outputs that are indicative of faults. These outputs, together with the expected output for a fault free circuit, are stored for use in the real circuit test mode. In the actual test mode an analogue input is applied to the circuit of Figure 1, as shown, and the digital output is compared with the stored outputs, thereby to determine whether there are any faults.

The approach of the Balivada article is fundamentally different to that of the present invention and relies completely on the transformation of the original analogue circuit to a digital equivalent. This is not necessary in the present invention. Instead, as specified in claim 1 the

digitised input test sequence is applied to the analogue/mixed signal circuit and not a digital equivalent. Furthermore, there is no mention in the Balivada article of any form of input sequence optimisation. Instead, it seems that an input sequence is merely selected and a series of digital outputs simulated based on the transformed digital equivalent of the analog block. There is no discussion of the significance of the input signal and absolutely no attempt made to optimize it. Hence, there is no disclosure in Balivada of inputting a digitised input test sequence to a fault free version of the analogue or mixed signal circuit or computer simulation thereof; inputting the same digitised input test sequence to a version of the circuit or computer simulation thereof that has at least one specified fault, and using an optimisation algorithm to vary the input test sequence; monitor the outputs of the fault free and faulty circuits for each test sequence and identify an optimised digitised input sequence. Furthermore, after the simulation is done according to the Balivada article, the test signal used is analog. Hence, there is no disclosure of using the digitised input to identify a corresponding *digitised output* for use in a test mode.

The problem with the disclosures of the Balivada article is that transforming the original analogue circuit to a Z-domain digital equivalent is very difficult and inevitably introduces errors. Hence, in practice, the results obtained are not good enough to test analogue circuits to a standard sufficient for manufacturing purposes. In addition, there is no appreciation of the sensitivity of the testing to the input signal and consequently no attempt made to optimize this.

The present invention overcomes these problems by optimising the input signal by applying a series of digitised test sequences directly to computer simulations of the analogue/mixed-signal circuit of interest and varying these using an optimisation algorithm until an optimum digitised input sequence is found. Then this optimised digitised input sequence is used to determine a corresponding digitised output sequence, and both the digitised input and output sequences are subsequently used in the test mode.

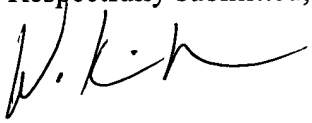
In light of the above, Applicants respectfully submit that the claims are patentable over the cited references.

CONCLUSION

In view of the amended claims and the remarks presented above, it is respectfully submitted that all of the present claims of the application are in condition for immediate allowance. It is therefore respectfully requested that a Notice of Allowance be issued. The Examiner is encouraged to contact Applicant's undersigned attorney to resolve any remaining issues in order to expedite examination of the present application.

It is not believed that extensions of time or fees for net addition of claims are required, beyond those that may otherwise be provided for in documents accompanying this paper. However, in the event that additional extensions of time are necessary to allow consideration of this paper, such extensions are hereby petitioned under 37 CFR § 1.136(a), and any fee required therefore (including fees for net addition of claims) is hereby authorized to be charged to Deposit Account No. 16-0605.

Respectfully submitted,

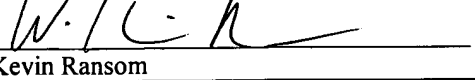


W. Kevin Ransom
Registration No. 45,031

Customer No. 00826
ALSTON & BIRD LLP
Bank of America Plaza
101 South Tryon Street, Suite 4000
Charlotte, NC 28280-4000
Tel Charlotte Office (704) 444-1000
Fax Charlotte Office (704) 444-1111

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to:
Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on June 28, 2006



W. Kevin Ransom

CLT01/4838980v1

D2

CRIS : A Test Cultivation Program for Sequential VLSI Circuits

Daniel G. Saab
Center For Reliable and High Performance Computing
Coordinated Science Laboratory
University of Illinois
Urbana, IL 61801

Youssef G. Saab
Department of Computer Science
University of Missouri
Columbia, Missouri

Jacob A. Abraham
Computer Engineering Research Center
University of Texas at Austin
Austin, TX 78758

ABSTRACT

This paper discusses a novel approach to cultivating a test for combinational and sequential VLSI circuits described hierarchically at the transistor, gate, and higher levels. The approach is based on continuous mutation of a given input sequence and on analyzing the mutated vectors for selecting the test set. The approach uses hierarchical simulation technique in the analysis to drastically reduce the memory requirement, thus allowing the test generation for large VLSI circuits. The algorithms are at the switch level so that general MOS digital designs can be handled, and both stuck-at and transistor faults are handle accurately. The approach has been implemented in a hierarchical test generation system, CRIS, that runs under UNIX on SPARC workstations. CRIS has been used successfully to generate tests with high fault coverage for large combinational and sequential circuits.

1. Introduction

The rapid increase in the level of integration in VLSI technology has made possible the implementation of larger and more complex chips. This complexity exceeds the limits of CAD tools which are crucial for the design of large circuits. Therefore, new methodologies must be developed to cope with the complexity and new design tools must be implemented.

Traditionally, test generation for digital circuits has been done at the gate-level [1-4], with failures approximated using the stuck-at-fault model. For sequential circuits, test generation techniques at the gate-level are based on constructing an iterative array model for the circuit [1-4]. However, it has been established that the gate-level, stuck-at fault model is not sufficient to describe the effects of physical failures [5]. In addition, for sequential circuits, the gate-level models do not consider any internal faults in the storage elements. Therefore, internal faults of sequential elements may or may not be detected by the derived test. Thus, the fault coverage reported by a gate-level test generator may be quite optimistic, and the actual quality of the tested product may not be as high as expected. These facts have spurred interest in performing test generation at the transistor level and in using alternatives to the stuck-at fault model [6-8]. While switch-level tests are very accurate in approximating the effects of failures and are effective for combinational circuits, they are ineffective for sequential VLSI circuits. In addition, they tend to require large amounts of memory and computer time since they have to deal with memory elements and bidirectional transistors. Thus, the increase in accuracy comes at the expense of higher memory and CPU cost.

The fault coverage of a proposed test sequence is usually determined by using fault simulation software packages. All test generation packages use fault simulation in the loop that identifies all faults detected by a proposed test to reduce the number of target faults. Test generation for gate level faults is itself quite complex; it is much more difficult to generate tests at the transistor level, and include timing and dynamic effects of realistic failures. Fault simulators can take the static tests and evaluate their effectiveness for realistic failures and dynamic behavior.

Ideas for exploiting the power of a fault simulator to derive tests to diagnose faults in asynchronous sequential circuits were proposed almost 30 years ago by Seshu [9]. Candidate tests were derived from previous tests by changing one input line at a time, and were evaluated using a fault simulator to determine their ability to distinguish between different faulty machines. More recently, Cheng and Agrawal [10] developed a test generator called CONTEST, which derives a new test vector by changing a bit in a current vector based on dynamic controllability and observability cost functions; a concurrent fault simulator is used to evaluate the derived test. Results showed that tests with good fault coverage were derived for circuits with up to 1539 gates and 73 flip-flops.

In this paper we present a new approach for test cultivation of multi-level of VLSI circuits which is based on ideas from genetic algorithms [11]. Although the approach is simulation-based and similar to [9] and [10], it uses much more elaborate (although computationally simple) techniques to generate new tests. The new approach has been implemented in a computer program, CRIS, which has been tested and verified on a switch level model of the ISCAS sequential benchmark circuits [12], the largest of which are an order of magnitude larger than the circuits reported by previous related techniques. Another key point of this work is its application to large MOS digital circuits with both transistor and stuck-at faults. We show that the technique can achieve high fault coverage for faults at the switch-level which results in high fault coverage for gate level faults.

The remainder of the paper is organized as follows: section 2 describes our basic approach to cultivating tests, section 3 details the various stages of the algorithm, and section 4 outlines our implementation and presents results.

2. Approach

Since conventional test generation algorithms perform poorly even on large combinational circuits, and since they cannot generate high quality tests for sequential circuits or for

† This work was supported in part by the Semiconductor Research Corporation Contract 91-DP-109 at the University of Illinois and in part by the Texas Advanced Research Program.

0-8186-3010-8/92 $03.00 © 1992 IEEE

realistic faults, such as transistor level and delay faults, we feel that a new approach needs to be considered for this problem. Our approach is based on two facts.

(1) Fault simulators can handle large sequential circuits, and deal with transistor level and delay faults, and therefore, any effective test generation technique should be able to exploit this capability.

(2) The search spaces in test generation are extremely large, and any effective technique should keep the history of some potentially points in the search space for later use

Our approach to deriving high quality tests for large sequential circuits is based on exploiting the power of a switch level fault simulator and on using genetic algorithms to guide the test generation procedure. Genetic search are very powerful for many optimization problems [11], and they have begun to be used in computer-aided design applications such as placement of modules on a VLSI chip [13]. The algorithms implicitly preserve the useful history during the search process by generating successive populations of possible solutions from the better solutions in a given population.

The search spaces involved in generating a test for a particular fault in a complex integrated circuit can be very large, and a brute force attempt to search it can literally take years of computational time. Genetic algorithms have shown the ability to search large spaces and move toward better solutions by selection of possible solutions from a large population. In order to apply genetic techniques to this problem we use an Objective function which is based on balanced circuit activity as recorded by logic simulator, and the extent to which errors due to faults have been propagated to primary outputs. Figure 1. shows a partitionned circuit with logic activity in each partition. As shown in Figure 1. The problem is to find candidates t_i to balance logic activities The activities is monitored during simulation, and used to rank and to select candidate vectors.

These candidate sequences are then modified using the classical techniques of reproduction, crossover and mutation [11]. During reproduction individual vectors or sequences with better objective functions are copied. Crossover, or splicing, involves taking substrings (from vectors) or subsequences (from sequences) and splicing them together to produce a new vector or sequence. Finally, mutation randomly flips bits in the vectors and sequences. The combined result of these operations is the production of new candidate sequences which, in turn, are evaluated by the fault simulator.

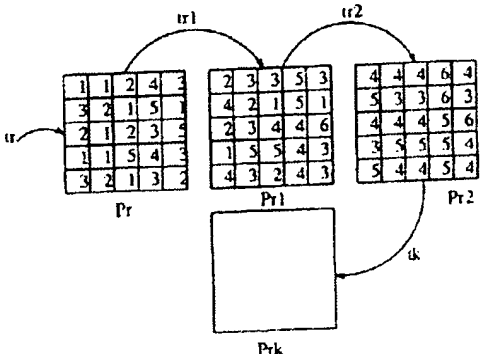


Figure 1. Desired activity profile

3. System Design

As described above, the objective of the test generation system is to produce a test sequence by modifying vectors from a candidate sequence. The tests should be for a sequential circuit, and should achieve high coverage for faults at both the gate and switch levels. Figure 2 gives a block diagram of the CRIS system which achieves these goals.

The initial test sequences may be generated randomly or be the set of functional or verification vectors produced by the designer. The key steps in producing the high quality tests are the mutation of the vectors and the analysis of the effectiveness of the resulting candidate sequences.

Fault simulation of a particular sequence identifies the set of hitherto undetected faults detected by the sequence, and provides a measure of the effectiveness of the sequence. Information about the activity of internal nodes during the fault simulation is collected, and points in the circuit where fault propagation was blocked are identified. This information is used to further characterize the candidate vectors as to how close they came to detecting faults, and their potential for detecting other faults if they are modified slightly.

The vectors are modified using the above date which involves *reproduction* (copying potentially useful candidate vectors and sequences), followed by either *mutation* (flipping bits in a vector) or *splicing* of vectors (producing a new vector using substrings from two other vectors) or sequences (producing a new sequence from subsequences of existing sequences). The information collected during simulation is used to identify bits for mutation or the vectors and sequences to splice together a new vector or sequence. The modification process is repeated until the desired fault coverage is achieved, a user-specified number of modifications have not resulted in further faults being detected, or a given CPU time limit has been exceeded. If the time limit is increased, CRIS will generate more candidate sequences; this approach, therefore, provides a mechanism to efficiently utilize the available CPU time.

To implement the above, circuit is partitioned into set of subcircuits. New vectors are generated from old one to produce new activity in a given subcircuit with low level of activity. The main algorithm selects a subcircuit s and perform either bit flipping *mutation*, *splicing* of two vectors, or *splicing* of sequences of vectors depending on the level of activity of subcircuit s. This process of subcircuit selection and vector

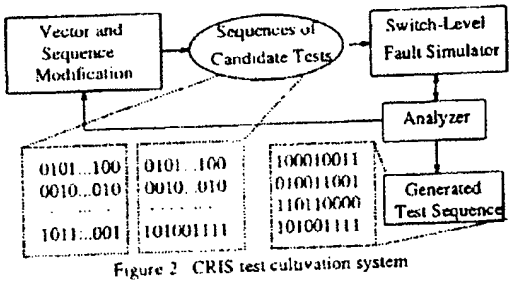


Figure 2 CRIS test cultivation system

modification is repeated until a desired coverage is achieved or until one of the stopping criteria mentioned above is met.

Procedure *splice* shown in Figure 3 does splicing on vectors. It marks useful parts of v1 and v2. Good part of v1 is substituted by part of v2 and vise versa. The other parts of v1 and v2 are copied at random. During the marking process, shown in Figure 4, if any vector causes activity in subcircuits that has low activity level this vector is saved (Cross-over) for further consideration at a later time.

Procedure *splice_vector* shown in Figure 5 does splicing on sequence of vectors. It identifies good vectors in V1 and V2. Good vector is a vector that causes activity in subcircuit *s*. Good vectors of V1 and V2 are appended to form a new sequence V. This sequence is the one used for fault simulation. During fault simulation if fault activity does not increase we might drop part of the sequence V

```
splice( v1, v2, S) {
    mark_good_bits( v1, s);
    mark_good_bits( v2, s);
    vk = combine( v1, v2 );
}
```

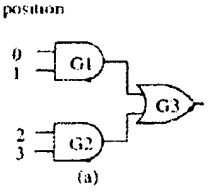
Figure 3. Procedure splice.

```
mark_good_bits( v, s ) {
    for( each bit b in v ) do {
        vk = v with bit b flipped ;
        n_event = logic_simulation( vk );
        if( n_event is high ) mark bit b as good;
        undo( vk );
    }
}
```

Figure 4. Procedure mark_good_bits.

```
splice_vector( V1, V2, s ){
    mark_good_vector( V1, s ),
    mark_good_vector( V2, s );
    V = combine_vectors( V1, V2 );
}
```

Figure 5. Procedure splice_vector.



(a)

vectors	inputs 0123	G1 S	G1 E	G2 S	G2 E	G3 S	G3 E
t0	1111	1	1	1	1	1	1
t1	0111	0	2	1	1	1	1
t2	1100	1	3	0	2	1	1
t3	0000	0	4	0	2	0	2
etc...							

(b)

Figure 6. (a) Simple example and (b) vector generated

Consider the example shown in Figure 6. Assume in this example that this circuit was not initialized and that an initial seed vectors of all ones and all zeros are given {Seed={(1111),(0000)}}. Initial vector t0 is picked from the initial seed at random and applied to the circuit. Next a subcircuit with low activity count is selected in this example gate G1 is selected. Assuming that we need to increase the event count by only one, two vectors v1=1111 and v2=0000 are selected for mutation. mark_good_bits(v1, G1) marks positions zero and one as good and adds vectors (1101) and (1110) to the seeds because they created activity at gate G2. mark_good_bits(v2, G2), also, marks positions 0 and 1 of v2 as being good and adds vectors (0010) and (0001) to the seeds because they created activity at gate G2. Then combine is called (t1 = Combines(v1,v2)) which returns a vector that is a combination of the good part of v1 and v2. If both good parts overlap, then corresponding bits are substituted at random (t1 = 0111). Fault simulation is called to process t1. G2 is selected next as target to increase activity. Vectors V1=0000 and v2=1101 are selected for mutation because they caused events at G2 before. The same steps done for G1 are repeated with new v1, v2, and G2. test vector t2 = 1100 is generated and vectors (1110,0010,0001,0101,1001,1111) are added to the seed. The same process is repeated for G3 with v1=0000 and v2=0010 the result of which is the generation of a new vector t3=0000. This process is repeated until a desired coverage is achieved or a stopping criteria is met.

4. Implementation

In order to demonstrate the effectiveness of CRIS on combinational circuits, we give a summary which contains the results of test cultivation on the ISCAS 85 benchmarks [14]. The circuits were simulated on a SUN SPARC SLC workstation. For example, Circuit c432 has 524 stuck-at faults, 519 of which were detected. The test cultivation process required a total of 3674 tries. Note that tests with high fault coverage were found in a relatively short time. Table II contains the result of applying CRIS to the ISCAS sequential benchmark circuits [12]. For example, Circuit s208 has 215 stuck-at faults, 131 of which were detected. The test cultivation process required a total of 1458 tries and 715 tests were generated. The CPU time for this circuit is 22.81 second. the number of tries is very reasonable for a fault coverage of 60.93 compared to 63.7 from deterministic test generator HITEC [2]. It should be noted that both fault coverage and number of tests generated were very close to HITEC, and were obtained in a very low run time. The total run time to generate all results in table II is less than 3.495 hours of CPU time on an SUN SLC workstation with 16 Meg. of memory. Only random

Ckt name	# of tries	# of tests	# of faults	detctd faults	Fault covrge	CPU (sec)
c432	3674	72	524	519	99.0	81.0
c499	3152	553	758	749	98.8	18.0
c880	5309	229	942	937	99.4	19.8
c1355	5543	205	1574	1556	98.8	58.0
c1908	4501	253	1879	1852	98.5	86.2
c2670	8000	398	2747	2290	83.3	263.7
c3540	8000	452	3428	3277	95.5	159.0
c5315	8000	682	5350	5258	98.2	426.3
c6288	2822	131	7744	7709	99.5	60.6
c7552	8000	672	7550	7120	94.3	554.7

Table I Results for ISCAS 85 combinational circuits.

seeds were used for this experiment. Seed extracted from design verification vectors are likely to produce even better coverage. For the ISCAS89 circuits the generated test were verified using VERIFAULT from CADANCE.

ckt name	#of try	#of tsts	# of faults	Flt Cvrg CRIS	Flt Cvrg HITEC	CPU sec CRIS
s208	1458	715	215	60.9	63.7	22.8
s298	922	476	308	82.1	86.0	16.2
s344	1312	115	350	93.7	95.5	19.9
s382	524	246	399	68.6	70.9	16.7
s386	2920	1230	384	76.0	81.7	16.9
s400	1061	758	421	84.7	76.0	35.3
s420	2348	857	430	33.7	41.3	65.4
s444	1079	519	474	83.7	63.0	34.2
s526	978	692	555	77.1	51.7	48.4
s641	8000	628	467	85.2	86.5	87.1
s713	6428	1124	581	81.7	81.9	94.9
s820	2439	1381	850	53.1	95.6	135.8
s832	2359	1328	870	42.5	93.9	105.1
s838	2664	1078	857	28.2	29.6	216.8
s1196	7644	2744	1242	95.0	99.7	97.0
s1238	7185	4313	1355	90.7	94.6	131.1
s1423	3444	2696	1515	77.0	37.1	580.5
s1488	3155	1960	1486	91.2	97.0	186.0
s1494	2947	1928	1506	90.1	96.4	171.2
s5378	3822	1255	4603	65.8	70.2	948.6
s35932	3407	1525	39094	88.2	89.3	9553.4

Table II. Results on the ISCAS 89 sequential circuits.

Result from running CRIS on a set of bit-serial FIR filters with 3 inputs and one output ranging from 4 bits 'fir4' to 32 bits 'fir32' is shown in Table III. Circuit 'fir4' contains 453 gate and 106 flip-flops. Circuit 'fir8' has 1720 gate and 355 flip-flops. Circuit 'fir16' has 3424 gate and 699 flip-flops. Circuit 'fir32' has 6928 gate and 1435 flip-flops. It is very hard to generate tests for such bit-serial circuits using traditional techniques. Again high fault coverage (close to 100% efficiency was achieved).

ckt name	# of tries	# of tests	# of faults	Fault Covrag	Eff	CPU (sec)
fir4	861	610	1141	75.28	98.2	93.7
fir8	1914	1643	3015	82.52	99.4	607.9
fir16	2250	2003	9114	82.63	99.3	462.1
fir32	2826	2592	18411	81.29	98.0	1512.3

Table III. Results on Bit Serial FIR filters

Table VI shows more results. Circuit 'add' is an eight bit adder containing 368 transistors, 144 stuck-at faults, and 568 transistor faults. When stuck-at faults were targeted, the test sequence obtained had a fault coverage of 72.9% for transistor faults and of 99% for stuck-at fault. On the hand, when targeting transistor faults the fault coverage was 76.26% for transistor faults and 100% for stuck-at faults.

ckt name	#of Trans	#of sa faults	#of Trans. faults	Stuck-at covrag	Trans. covrag
add	368	144	568	99.0	72.9
cbcs	333	482	620	100.0	70.9
PLA	4096	3004	4095	99.8	70.9
mult8	3328	1408	6272	97.3	72.8

Table VI. Results when targeting stuck-at faults.

ckt name	#of Trans	#of sa faults	#of Trans. faults	Stuck-at covrag	Trans. covrag
add	368	144	568	100.0	76.2
cbcs	333	482	620	100.0	75.9
PLA	4096	3004	4095	100.0	81.3
mult8	3328	1408	6272	99.9	82.3

Table VI. (Cont.) Results when targeting transistor faults.

5. References

[1] H-K. T. Ma, S. Devadas, A. R. Newton, and A. Sangiovani-Vicentelli, "Test Generation for Sequential Circuits," *IEEE, Transactions on Computer Aided-Design.*, vol. CAD-7, pp. 1081-1093, Oct. 1988.

[2] T. M. Niermann and J. H. Patel, "HITEC: A Test Generation Package for Sequential Circuits," *European Design Automation Conference*, pp. 214-218, 1991.

[3] W.-T. Cheng, "The BACK Algorithm for Sequential Test Generation," *International Conference on Computer Aided Design*, pp. 214-218, 1991.

[4] Miron Abramovici, Melvin A. Breuer, and Arthur D. Friedman, *Digital Systems Testing and Testable Design*. NewYork: Computer Science Press, 1990.

[5] J. Galiay et al., "Physical Versus Logic Fault Models in MOS LSI Circuits, Impact on their Testability," *Int. Symp. Fault Tolerant Computing*, pp. 195-202, 1979..

[6] R. L. Wadsack, "Fault Modeling and Simulation of CMOS and MOS Integrated Circuits," *The Bell System Tech. Journal*, pp. 1449-1474, June 1978..

[7] Y. M. El-Ziq, "Automatic Test Generation for Stuck-Open Faults in CMOS VLSI," *Design Automation Conference*, pp. 347-352, June 1981..

[8] M. K. Reddy, S. M. Reddy, and P. Agrawal, "Transistor Level Test Generation for MOS Circuits," *Design Automation Conference*, pp. 825-828, 1985..

[9] S. Seshu and D. N. Freeman, "The diagnosis of asynchronous sequential switching systems," *IRE Transactions on Electronic Computing*, vol. EC-11, pp. 459-465, August 1962.

[10] K.-T. Cheng and V. D. Agrawal, *Unified Methods for VLSI Simulation and Test Generation*. Boston, MA: Kluwer Academic Publishers, 1989.

[11] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Massachuseus: Addison-Wesley, 1989.

[12] F. Brglez, D. Bryan, and K. Kozminski, "Combinational Profiles of Sequential Benchmark Circuits," in *Proceedings of the 1989 Int. Symp. on Circuits and Systems*, Portland, Oregon, May 1989.

[13] J. P. Cohoon and W. D. Paris, "Genetic Placement," in *Proc. of the IEEE International Conference on Computer-Aided Design*, Santa Clara, CA, pp. 422-425, November 1986.

[14] F. Brglez and H. Fujiwara, "A Neutral Netlist of 10 Combinational Benchmark Circuits and a Target Translator in Fortran," *Proc. of the Int. Test Conf.*, pp. 785-794, 1985.